НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Навчально-науковий інститут комп’ютерних інформаційних технологій

Кафедра комп’ютеризованих систем управління

ЗВІТ

про виконання домашнього завдання

з дисципліни: «Програмування»

тема роботи: Пошагова стратегія

Роботу виконав: студент групи СП-125

Козлов Олексій Вячеславович

Роботу перевірив:Старший викладач Станко С.М.

Київ – 2018

**Зміст**

1. Вступ
2. Призначення та область застосування
3. Постановка задачі
4. Використовувані методи
5. Опис алгоритм програми з описом функцій складових частин і зв’язки між ними
6. Опис вхідних та вихідних даних
7. Програмне і апаратне забезпечення програми
8. Керівництво користувача
9. Висновок
10. Джерела
11. Додаток А лістинг програми
12. Додаток Б результати роботи програми

**Вступ**

**Завдання роботи**

Полягає в написанні програми, що дозволяє двом гравцям створювати армії та воювати між собою мовою програмування С.

Програма має працювати з файлами і бути кросплатформеною, сам код має містити не менше 150 строк.

**Методи і засоби**

1. Армії мають містити юнітів для їх збереження по за програмою використовується файл, з цього ж файлу вони будуть зчитуватись.
2. Для збереження армій під час роботи програми використовується масив структур
3. Для реалізації ігрового поля використовується чотиринаправленний список
4. Для реалізації ігрових функцій використовуються різні операції над вказівниками на структури

**Очікувані результати**

1. Виконання усіх умов задачі
2. Поглиблення знань з створення програмних продуктів
3. Поглиблення знань з використання структур, вказівників, динамічної пам’яті і операцій з файлами

**Призначення та область застосування**

**Теоретична**

Фрагменти коду можна використовувати як приклад застосування тих чи інших операторів

**Практична**

Програму можна використовувати в розважальних цілях, для покращення стратегічних навичок гравців та для розвитку їх логічного мислення

**Наукова**

Оптимізація операцій с файлами і динамічною пам’яттю

**Постановка задачі**

1. Опис структур unit і field
2. Створення масиву структур unit і його заповнення
3. Створення можливості додавати нових юнітів в файл і зчитувати іх
4. Створення списку із структур field і його ініціалізація
5. Створення можливості пересувати юнітів по полю і їх взаємодії
6. Забезпечення повного ігрового процесу
7. Створення умови виходу із програми
8. Забезпечення кросплатформеності програми

**Використовувані методи**

1. Стандартні умовні оператори if else , switch case
2. Оператор циклів for fopen, fclose
3. Функції роботи з файлами
4. Функції вводу і виводу printf, fprintf, scanf, fscanf, fputc, getc, getchar()
5. Функції роботи з пам’яттю malloc, free
6. Директиви препроцесорної обробки #include, #define, #ifdef, #else, #endif

**Опис алгоритму програми з описом функцій складових частин і зв'язки між ними**

Початок

main

key\_exit

key\_end\_game

turn\_num

UNITS

CELLS

key\_do=0

Кінець

key\_do

так

key\_do=1

ні

Unit\_Create

так

key\_do=2

ні

Print\_all\_units

ні

key\_do=3

так

Cr\_field(&head,&last,CELLS)

Print\_all\_units

Units(un\_ptr,UNITS,1)

Units(un\_ptr,UNITS,2)

Place\_un(un\_ptr,&head,UNITS,CELLS)

for(;key\_end\_game==0&&turn\_num<50;turn\_num++)

Show\_field(&head,CELLS)

Army(un\_ptr,UNITS,(turn\_num&1)+1)

Move\_un(un\_ptr,&head,CELLS,(turn\_num&1)+1)

key\_end\_game=win\_ch(un\_ptr,UNITS)

Початок

Cr\_field

\*\*head

\*\*last

cells

\*current

for(int i=0;i<cells\*cells;i++)

так

Кінець

(current=(field \*)malloc(sizeof(field)))=NULL

ні

current->next=NULL

current->prev=NULL

current->up=NULL

current->down=NULL

current->un=NULL

current->x=1+i%cells

current->y=1+i/cells

ні

так

!(\*head)

(\*last)->next=current

\*head=current

\*last=current

,,

ні

current->y!=1

так

\*curr

curr=current->prev

curr->x!=current->x

curr=curr->prev

curr->down=current

current->up=curr

Show\_field

Початок

\*\*head

cells

\*current

current=\*head

i

for(int i=1;i<=cells;i++)

for(int i=0;i<cells;i++)

Кінець

ні

current->x=1

current->y

так

for(int i=0;i<cells;i++)

(current->un)->mark

так

ні

current->un!=NULL

current=current->next

Unit\_Create

Початок

\*dt

dt=fopen("units.txt","a")

так

dt=NULL

ні

new\_unit

new\_unit.mark

new\_unit.at

new\_unit.hp

new\_unit.move

fclose(dt)

Кінець

dt=NULL

dt=fopen("units.txt","a")

\*dt

Початок

Print\_all\_units

ні

так

i=1

for(char ch='\n';ch!=EOF;ch=getc(dt))

fclose(dt)

ch

ні

ch='\n'

Кінець

так

i

Початок

Char\_to\_int

ch

.

так

2

ch=’2’

ні

1

ch=’1’

ні

так

ні

ch=’3’

3

ні

ch=’4’

4

так

так

6

ch=’6’

ні

так

5

ch=’5’

ні

так

ні

так

ch=’7’

7

ні

так

ch=’8’

8

9

ch=’9’

так

ні

0

Кінець

Read\_unit

Початок

\*un\_ptr

unit\_num

\*dt

dt=fopen("units.txt","r")

так

dt=NULL

ні

for(int i=1;i<unit\_num;i++)

for(ch=getc(dt);ch!='\n'&&ch!=EOF;)

ch=getc(dt)

ch=getc(dt)

un\_ptr->mark=ch

ch=getc(dt)

un\_ptr->at=Char\_to\_int(ch=getc(dt))

ch=getc(dt)

un\_ptr->hp=Char\_to\_int(ch=getc(dt))

ch=getc(dt)

un\_ptr->move=Char\_to\_int(ch=getc(dt))

fclose(dt)

Кінець

Units

Початок

\*un\_ptr

num\_un

player

for(int i=player-1;i<num\_un\*2;i+=2)

Кінець

unit\_num

Read\_unit((un\_ptr+i),unit\_num)

Початок

Army

\*un\_ptr

num\_un

player

n=1

for(int i=player-1;i<num\_un\*2;i+=2)

Кінець

n

(un\_ptr+i)->mark

(un\_ptr+i)->at

(un\_ptr+i)->hp

(un\_ptr+i)->move

n++

win\_ch

Початок

\*un\_ptr

num\_un

def\_p1=0

def\_p2=0

for(int i=0;i<num\_un\*2;i++)

так

ні

i&1

ні

(un\_ptr+i)->hp!=0

(un\_ptr+i)->hp!=0

так

так

def\_p2=1

def\_p1=1

так

!def\_p1&&!def\_p2

!def\_p1

ні

так

1

1

!def\_p2

так

ні

ні

1

0

Кінець

Початок

Att

\*un\_ptr

\*un\_ptr1

\*un\_ptr2

player

\*current

(un\_ptr1-un\_ptr2)%2!=0

ні

0

un\_ptr2->hp-=un\_ptr1->at

так

ні

un\_ptr2->hp<=0

un\_ptr2->hp=0;

current->un=NULL;

так

1

Кінець

Place\_un

Початок

\*un\_ptr

\*\*head

num\_un

cells

\*current=\*head

так

num\_un>cells

num\_un=cells

ні

Кінець

for(int i=0;i<num\_un\*2;i+=2)

for(int j=0;j<cells-1;j++)

current->un=(un\_ptr+i)

current=current->next

current->un=(un\_ptr+i+1)

current=current->next

Початок

Move\_un

\*un\_ptr

\*\*head

cells

player

unit\_tu

\*curr\_un

curr\_un=(un\_ptr+(player-1)+2\*(unit\_tu-1))

ні

unit\_tu>cells||curr\_un->hp=0

for(int i=player-1;curr\_un->hp=0&&i<cells\*2;i+=2)

так

curr\_un=(un\_ptr+i)

\*current=\*head

for(int i=0;current->un!=curr\_un&&i<cells\*cells;i++)

current=current->next

for(int val=1;val==1;)

dx

dy

1

val=0

ні

(dx<1||dx>cells)||(dy<1||dy>cells)

так

val=1

1

key\_at=0

3

for(int mp=curr\_un->move;mp>0&&key\_at==0;mp--)

Кінець

ні

так

current->x<dx

так

current->next->un=NULL

ні

key\_at=Att(un\_ptr,curr\_un,current->next->un,player,current->next)

current->prev->un=current->un

current->un=NULL

current=current->prev

так

ні

current->x>dx

так

current->prev->un=NULL

key\_at=Att(un\_ptr,curr\_un,current->prev->un,player,current->prev))

ні

current->prev->un=current->un

current->un=NULL

current=current->prev

2

2

ні

так

current->y>dy&&key\_at=0

так

current->up->un=NULL

ні

key\_at=Att(un\_ptr,curr\_un,current->up->un,player,current->up)

current->up->un=current->un

current->un=NULL

current=current->up

ні

так

current->y<dy&&key\_at=0

current->down->un=NULL

так

key\_at=Att(un\_ptr,curr\_un,current->prev->un,player,current->prev))

ні

current->down->un=current->un

current->un=NULL

current=current->down

на ст. 17

3

**Опис вхідних та вихідних даних**

В програмі використовуються різні типи даних: int, unsigned int, char, string. Користувач вводить тільки int і string. Під час виконання програми користувач може побачити різні набори з символів і цифр також одна з функцій записує введені користувачем значення в файл.

**Програмне і апаратне забезпечення програми**

Програма запускається в консолі, відеократи не потребує, Операційні системи: Windows і Linux, Exe займає 160 КБ (163 840 байт) на диску.

**Керівництво користувача**

Для успішного виконання програми користувачу потрібно вводити данні відповідно до запиту програми.

Показник рухомості значить що юніт може рухатись в будь-якому напрямку на N клітинок. Рух і атака по осі х має пріоритет.

Гра закінчується коли в одного з гравців нема живих юнітів або через 50 ходів від початку гри.

**Висновок**

Програму завершено та протестованно. Всі поставлені задачі було виконано в повній мірі

**Література**

МакГрат М. Программирование на С для начинающих - М.: Эксмо, 2016 - 192 с.

Кочан С. Программирование на языке С. - М.: ООО «И.Д. Вильямс», 2007. - 496 с.

Керниган Б., Ритчи Д. Язык программирования C - М.: Диалектика-Вильямс, 2015. - 288 с.

Перри Г., Миллер Д. Программирование на С для начинающих - М.: Эксмо, 2015. - 368 с.

**ДОДАТОК А**

dzmain.c

#include<stdio.h>

#include"dz\_head\_field.h"

#include"dz\_head\_unit.h"

#include"dz\_head\_files.h"

#include"dz\_head\_units\_and\_field.h"

#define UNITS 5%10//кількість юнітів

#define CELLS 9%10//довжина і ширина ігрового поля

struct Unit\_u//структура юнітів

{

char mark;//позначення персонажа

unsigned int at;//показник атаки персонажа

int hp;//показник здоров'я персонажа

unsigned int move;//показник рухомості персонажа

};

typedef struct Unit\_u unit;

struct Field\_u//структура поля

{

unsigned int x;//координата х

unsigned int y;//координата у

struct Field\_u \*next;//наступна клітинка

struct Field\_u \*prev;//попередня клітинка

struct Field\_u \*up;//верхня клітинка

struct Field\_u \*down;//нижня клітинка

unit \*un;//юніт в клітинці

};

typedef struct Field\_u field;

unit units[UNITS\*2];//массив юнітів

unit \*un\_ptr=units;//вказівник на массив юнітів

int main()

{

#ifdef \_WIN32//Препроцесорна логічна умова для Windows системи

system("cls");

#endif

unsigned int key\_exit=0;//ключ виходу із програми

unsigned int key\_do=0;//ключ дії

printf("\nEnter a key to do smth\n1st to create new unit\n2nd to view all units\n3rd to start game\nOr other to exit\n");

for(;key\_exit==0;)//цикл роботи програми

{

printf("\tkey = ");

scanf("%d",&key\_do);//введення ключа дії

switch (key\_do)

{

case 1://створення нового юніта

{

Unit\_Create();

break;

}

case 2://виведення всіх юнітів

{

Print\_all\_units();

break;

}

case 3://гра

{

field \*head=NULL;//Початок списка (ігрового поля)

field \*last=NULL;//Кінець списка (ігрового поля)

unsigned int key\_end\_game=0;//ключ виходу з гри

unsigned int turn\_num=0;//номер ходу

Cr\_field(&head,&last,CELLS);//створення поля

Print\_all\_units();//виведення всіх юнітів

printf("Chose units player №1\n");

Units(un\_ptr,UNITS,1);//армія 1го гравця

printf("Chose units player №2\n");

Units(un\_ptr,UNITS,2);//армія 2го гравця

Place\_un(un\_ptr,&head,UNITS,CELLS);//розміщення юнітів на полі

for(;key\_end\_game==0&&turn\_num<50;turn\_num++)//цикл гри

{

Show\_field(&head,CELLS);//виведення ігрового поля

printf("Player #%d turn\n",(turn\_num&1)+1);

Army(un\_ptr,UNITS,(turn\_num&1)+1);//виведення армії гравця

Move\_un(un\_ptr,&head,CELLS,(turn\_num&1)+1);//хід гравця

key\_end\_game=win\_ch(un\_ptr,UNITS);//перевірка закінчення гри

}

printf("Number of moves = %d\n",turn\_num);//кількість ходів

#ifdef \_WIN32

free(head);//звільнення пам'яті

free(last);//звільнення пам'яті

#endif

break;

}

default://вихід з програми

{

key\_exit=1;

}

}

}

getchar();

getchar();

return 0;

}

dzfield.c

#include<stdio.h>

#include<malloc.h>

struct Unit\_u

{

char mark;//позначення персонажа

unsigned int at;//показник атаки персонажа

int hp;//показник здоров'я персонажа

unsigned int move;//показник рухомості персонажа

};

typedef struct Unit\_u unit;

struct Field\_u

{

unsigned int x;//координата х

unsigned int y;//координата у

struct Field\_u \*next;//наступна клітинка

struct Field\_u \*prev;//попередня клітинка

struct Field\_u \*up;//верхня клітинка

struct Field\_u \*down;//нижня клітинка

unit \*un;//юніт в клітинці

};

typedef struct Field\_u field;

void Cr\_field(field \*\*head,field \*\*last,unsigned int cells)//функція створення поля

{

field \*current;

for(int i=0;i<cells\*cells;i++)//цикл формування поля заданого розміру

{

if((current=(field \*)malloc(sizeof(field)))==NULL)//перевірка наявності пам'яті

{

perror("no free memory\n");

}

else

{

current->next=NULL;//ініціалізація по замовченню

current->prev=NULL;

current->up=NULL;

current->down=NULL;

current->un=NULL;

current->x=1+i%cells;

current->y=1+i/cells;

if(!(\*head))//вказівник на перший елемент

\*head=current;

else

(\*last)->next=current;//вказівник на наступний елемент

if(\*last)

current->prev=\*last;//вказівник на попередній елемент

\*last=current;

if(current->y!=1)//якщо обраобляеться не перший рядок

{

field \*curr;//вказівник на верхній елемент

curr=current->prev;

for(;curr->x!=current->x;)

curr=curr->prev;

curr->down=current;//ініціалізація відповідних полів

current->up=curr;

}

}

}

}

void Show\_field(field \*\*head,unsigned int cells)//функція виводу поля

{

printf("\tField\n");

field \*current;

current=\*head;

printf("\t x|");

for(int i=1;i<=cells;i++)

printf(" %d |",i);

printf("\n\ty ");

for(int i=1;i<=cells\*4-1;i++)

printf("\_");

for(int i=0;i<cells;i++)//виведення елементів

{

if(current->x==1)//виведення першого елемента в рядку

{

printf("\n");

printf("\t%d |",current->y);

if(current->un!=NULL)

printf("%c |",(current->un)->mark);

else

printf(" |");

current=current->next;

}

for(int i=0;i<cells-1;i++)//виведення наступних

{

if(current->un!=NULL)

printf("%c |",(current->un)->mark);

else

printf(" |");

current=current->next;

}

printf("\n\t ");//останняй рядок

printf("|\_\_\_|");

for(int i=0;i<cells-1;i++)

printf("\_\_\_|");

}

printf("\n");

}

dzfiles.c

#include<stdio.h>

struct Unit\_u

{

char mark[2];//позначення персонажа

unsigned int at;//показник атаки персонажа

int hp;//показник здоров'я персонажа

unsigned int move;//показник рухомості персонажа

};

typedef struct Unit\_u unit;

void Unit\_Create()//функція створення юнітів

{

FILE \*dt;

dt=fopen("units.txt","a");//файл запису юнітів

if(dt==NULL)//якщо нема доступу

perror("no access to the file\n");

else

{//введення і запис характеристик

unit new\_unit;

printf("Enter the character designation (1 character): ");

scanf("%s",new\_unit.mark);

fprintf(dt,"\n");

fputc(\*new\_unit.mark,dt);

fprintf(dt,"|");

printf("Enter the attack rate of the character: ");

scanf("%d",&new\_unit.at);

fprintf(dt,"%d|",new\_unit.at%10);

printf("Enter the health rate of the character: ");

scanf("%d",&new\_unit.hp);

fprintf(dt,"%d|",new\_unit.hp%10);

printf("Enter the mobility rate of the character: ");

scanf("%d",&new\_unit.move);

fprintf(dt,"%d|",new\_unit.move%10);

fclose(dt);//закриття файлу

}

}

void Print\_all\_units()//виведення віх юнітів з файлу

{

FILE \*dt;

dt=fopen("units.txt","r");

if(dt==NULL)//якщо нема доступу

perror("no access to the file\n");

else

{//виведення юнітів

printf("\tAll units\n");

printf("\t \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

printf("\t|\t#\t| Mark | Attack | Health | Mobility |");

int i=1;

for(char ch='\n';ch!=EOF;ch=getc(dt))//посимвольне считування

{

if(ch=='|')//наступна характеристика

{

for(int j=0;j<10;j++)

printf(" ");

}

printf("%c",ch);

if(ch=='\n')//новий юніт

{

printf("\t|\t%d\t|",i);

i++;

}

}

printf("\n\t|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\n");

fclose(dt);//закриття файлу

}

}

dzhelp.c

int Char\_to\_int(char ch)//функція перетворення з char в int

{

switch (ch)

{

case '1':

return 1;

break;

case '2':

return 2;

break;

case '3':

return 3;

break;

case '4':

return 4;

break;

case '5':

return 5;

break;

case '6':

return 6;

break;

case '7':

return 7;

break;

case '8':

return 8;

break;

case '9':

return 9;

break;

default:

return 0;

break;

}

}

dzunit.c

#include<stdio.h>

#include"dz\_head\_help.h"

struct Unit\_u

{

char mark;//позначення персонажа

unsigned int at;//показник атаки персонажа

int hp;//показник здоров'я персонажа

unsigned int move;//показник рухомості персонажа

};

typedef struct Unit\_u unit;

void Read\_unit(unit \*un\_ptr,unsigned int unit\_num)//считування юніта з файлу

{

FILE \*dt;

dt=fopen("units.txt","r");

if(dt==NULL)//якщо нема доступу до файлу

perror("no access to the file\n");

else

{//считування характеристик

char ch;

for(int i=1;i<unit\_num;i++)//перехід на строку з потрібним юнітом

for(ch=getc(dt);ch!='\n'&&ch!=EOF;)//до кінця строки або файлу

ch=getc(dt);

ch=getc(dt);

un\_ptr->mark=ch;//запис позначення юніта

ch=getc(dt);

un\_ptr->at=Char\_to\_int(ch=getc(dt));//запис перетвореної в ціле число атаки юніта

ch=getc(dt);

un\_ptr->hp=Char\_to\_int(ch=getc(dt));//запис перетвореної в ціле число здоров'я юніта

ch=getc(dt);

un\_ptr->move=Char\_to\_int(ch=getc(dt));//запис перетвореної в ціле число рухомості юніта

fclose(dt);

}

}

void Units(unit \*un\_ptr,unsigned int num\_un,unsigned int player)//створення армії гравця

{

for(int i=player-1;i<num\_un\*2;i+=2)//цикл створення армії

{

printf("Enter the unit number to add him to the army: ");

int unit\_num;

scanf("%d",&unit\_num);//номер юніта в списку

Read\_unit((un\_ptr+i),unit\_num);//додання його до армії

}

}

void Army(unit \*un\_ptr,unsigned int num\_un,unsigned int player)//виведення армії гравця

{

printf("\tPlayer %d army \n",player);

printf("\t \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

printf("\t|\t#\t| Mark | Attack | Health | Mobility |");

int n=1;

for(int i=player-1;i<num\_un\*2;i+=2)//цикл виведення армії

{

printf("\n\t|\t%d\t|%c |%d |%d |%d |",n,(un\_ptr+i)->mark,(un\_ptr+i)->at,(un\_ptr+i)->hp,(un\_ptr+i)->move);//характеристики

n++;

}

printf("\n\t|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_|\n");

}

int win\_ch(unit \*un\_ptr,unsigned int num\_un)//перевірка закінчення гри

{

int def\_p1=0;//поразка гравця 1

int def\_p2=0;//поразка гравця 2

for(int i=0;i<num\_un\*2;i++)//перевірка наявності живих юнітів

{

if(i&1)//юніти другого гравця

{

if((un\_ptr+i)->hp!=0)//якщо нема живих

def\_p2=1;

}

else//юніти першого гравця

{

if((un\_ptr+i)->hp!=0)//якщо нема живих

def\_p1=1;

}

}

if(!def\_p1&&!def\_p2)//якщо в обох гравців нема живих юнітів

{

printf("Game over. \nResult: Draw\n");

return 1;

}

else if(!def\_p1)//перемога гравця 2

{

printf("Game over. \nResult: The winner is player #2\n");

return 1;

}

else if(!def\_p2)//перемога гравця 1

{

printf("Game over. \nResult: The winner is player #1\n");

return 1;

}

else//гра не закінчена

return 0;

}

dzunitandfield.c

#include<stdio.h>

struct Unit\_u

{

char mark;//позначення персонажа

unsigned int at;//показник атаки персонажа

int hp;//показник здоров'я персонажа

unsigned int move;//показник рухомості персонажа

};

typedef struct Unit\_u unit;

struct Field\_u

{

unsigned int x;//координата х

unsigned int y;//координата у

struct Field\_u \*next;//наступна клітинка

struct Field\_u \*prev;//попередня клітинка

struct Field\_u \*up;//верхня клітинка

struct Field\_u \*down;//нижня клітинка

unit \*un;//юніт в клітинці

};

typedef struct Field\_u field;

int Att(unit \*un\_ptr,unit \*un\_ptr1,unit \*un\_ptr2,unsigned int player,field \*current)//функція атаки

{

if((un\_ptr1-un\_ptr2)%2!=0)//якщо атакуючий і атакований не належать до одному гравцю

{

printf("Enemy takes %d damage\n",un\_ptr1->at);

un\_ptr2->hp-=un\_ptr1->at;//нанесення пошкоджень

if(un\_ptr2->hp<=0)//якщо атакований загинув

{

un\_ptr2->hp=0;

current->un=NULL;

printf("Enemy unit died\n");

}

return 1;//атака відбулася

}

else

return 0;//атака не відбулася

}

void Move\_un(unit \*un\_ptr,field \*\*head,unsigned int cells,unsigned int player)//функція руху юніта

{

printf("Choose unit for turn: ");

unsigned int unit\_tu=0;

scanf("%d",&unit\_tu);

unit \*curr\_un=(un\_ptr+(player-1)+2\*(unit\_tu-1));//выбір юніта для ходу

if(unit\_tu>cells||curr\_un->hp==0)//якщо цього юніта нама на полі

{

for(int i=player-1;curr\_un->hp==0&&i<cells\*2;i+=2)//обрання першого живого юніта

curr\_un=(un\_ptr+i);

printf("This is absent. \nFirst live unit is chosen for turn\n");

}

field \*current=\*head;

for(int i=0;current->un!=curr\_un&&i<cells\*cells;i++)//поточне положення юніта

current=current->next;

unsigned int dx,dy;//кординати точки призначення

printf("Enter coordinates x,y to move and / or attack: \n");

for(int val=1;val==1;)//поки координати не буде введено правильно

{

printf("\tx = ");//введення координат

scanf("%d",&dx);

printf("\ty = ");

scanf("%d",&dy);

val=0;

if((dx<1||dx>cells)||(dy<1||dy>cells))

{

val=1;

printf("Enter the coordinates correctly\n");

}

}

unsigned int key\_at=0;//наявність атаки при русі

for(int mp=curr\_un->move;mp>0&&key\_at==0;mp--)//цикл руху і атаки

{

if(current->x<dx)//якщо юніт зліва від точки призначення

{

if(current->next->un==NULL)//якщо наступна клітинка пуста

{

current->next->un=current->un;//рухатись

current->un=NULL;

current=current->next;

}

else

key\_at=Att(un\_ptr,curr\_un,current->next->un,player,current->next);//атакувати

}

else if(current->x>dx)//якщо юніт справа від точки призначення

{

if(current->prev->un==NULL)//якщо наступна клітинка пуста

{

current->prev->un=current->un;//рухатись

current->un=NULL;

current=current->prev;

}

else

key\_at=Att(un\_ptr,curr\_un,current->prev->un,player,current->prev);//атакувати

}

if(current->y>dy&&key\_at==0)//якщо юніт знизу від точки призначення

{

if(current->up->un==NULL)//якщо наступна клітинка пуста

{

current->up->un=current->un;//рухатись

current->un=NULL;

current=current->up;

}

else

key\_at=Att(un\_ptr,curr\_un,current->up->un,player,current->up);//атакувати

}

else if(current->y<dy)//якщо юніт зверху від точки призначення

{

if(current->down->un==NULL)//якщо наступна клітинка пуста

{

current->down->un=current->un;//рухатись

current->un=NULL;

current=current->down;

}

else

key\_at=Att(un\_ptr,curr\_un,current->down->un,player,current->down);//атакувати

}

}

}

void Place\_un(unit \*un\_ptr,field \*\*head,unsigned int num\_un,unsigned int cells)//розміщення юнітів на полі

{

field \*current=\*head;

if (num\_un>cells)//якщо кількість юнітів більша за кількість клітинок в одному столбці

num\_un=cells;//юніти займають весь стовбчик

for(int i=0;i<num\_un\*2;i+=2)//цикл розміщення по одному ряду

{

current->un=(un\_ptr+i);//розмістити в першому стовпці

for(int j=0;j<cells-1;j++)

current=current->next;

current->un=(un\_ptr+i+1);//розмістити в останьому стовпці

current=current->next;

}

}

dz\_head\_field.h

void Cr\_field(struct Field\_u \*\*head,struct Field\_u \*\*last,unsigned int cells);//створення ігрового поля

void Show\_field(struct Field\_u \*\*head,unsigned int cells);//виведення ігрового поля на екран

dz\_head\_files.h

void Unit\_Create();//Введення в файл. Створення юніту

void Print\_all\_units();//Виведення з файлу. Виведення всіх юнітів на екран

dz\_head\_help.h

int Char\_to\_int(char ch);//Перетворення з char в int

dz\_head\_unit.h

void Units(struct Unit\_u \*un\_ptr,unsigned int num\_un,unsigned int player);//Формування армії

void Army(struct Unit\_u \*un\_ptr,unsigned int num\_un,unsigned int player);//Виведення армії

int win\_ch(struct Unit\_u \*un\_ptr,unsigned int);//Перевірка завершення гри

dz\_head\_unit\_and\_field.h

void Move\_un(struct Unit\_u \*un\_ptr,struct Field\_u \*\*head,unsigned int cells,unsigned int player);//Переміщення

void Place\_un(struct Unit\_u \*un\_ptr,struct Field\_u \*\*head,unsigned int num\_un,unsigned int cells);//Розмащення юнітів на полі

**ДОДАТОК Б**

